AON: Towards Arbitrarily-Oriented Text Recognition

Zhanzhan Cheng, Yangliu Xu, Fan Bai, Yi Niu,
Shiliang Pu, Shuigeng Zhou

Presented by Shih-Ming Wang
ComputerVision Lab, UCSC

10-24-2018

Motivation

OCR in Practice

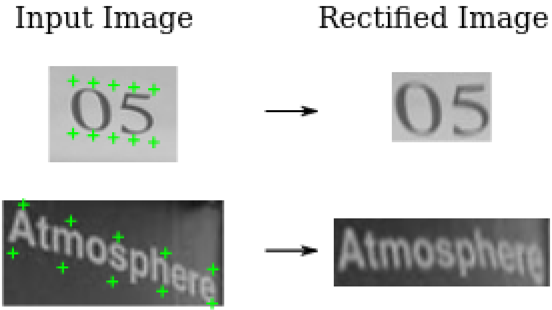
- Uneven lighting, blurring
- Perspective distortion
- Orientation
- Most traditional OCR system deals with regular tightly-bounded, horizontal texts

Previous Work I

Spatial Transformer Network Based Model

- The model learns to rectify input image by learn to translate and rotate, etc.
- Hard to optimize transformation network without geometric groundtruth
- Requires tricks on initialisation of model weights to guarantee training convergence

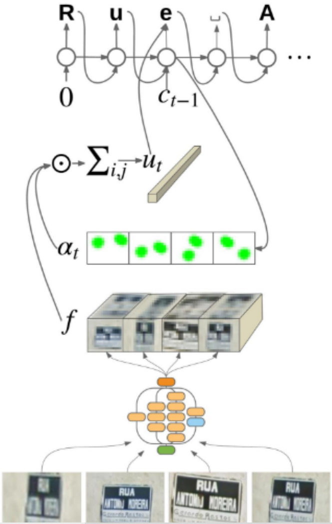
Previous Work II



Previous Work III

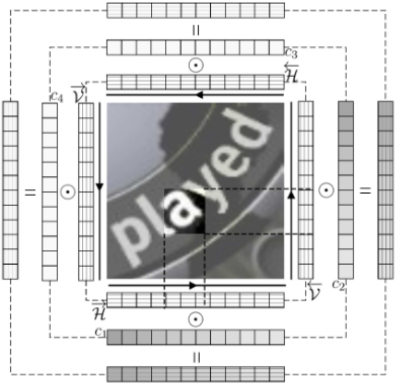
Attention Based Model

- Encode image into featuremap and use RNN to predict character sequences.
- Does not work well when directly applied to irregular texts



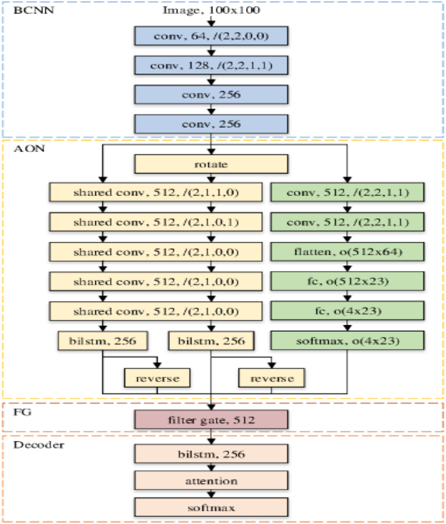
Method Intuition

- Extract feature of four direction and learn to weight them properly



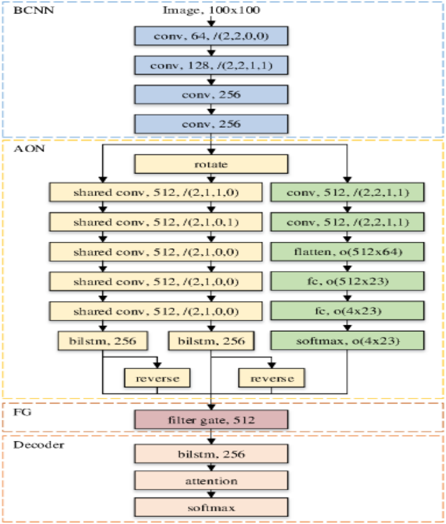
Architecture Overview

- Basal CNN(BCNN): extract low level image features



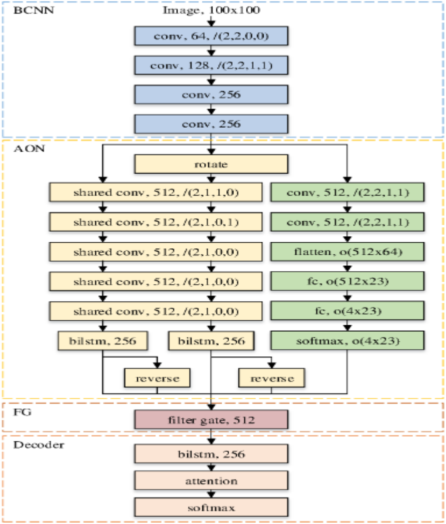
Architecture Overview

- Basal CNN(BCNN): extract low level image features
- Arbitrary(AON): extract high level features in 4 direction and calculate character placement clue



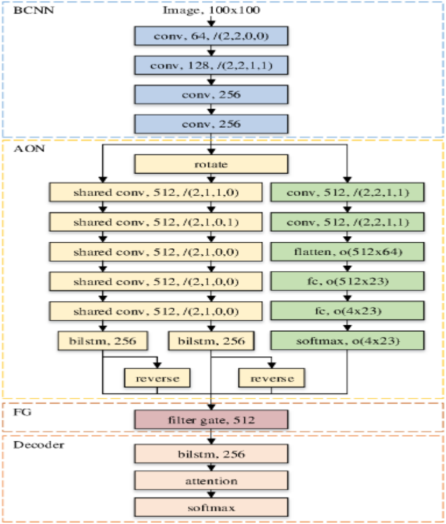
Architecture Overview

- Basal CNN(BCNN): extract low level image features
- Arbitrary(AON): extract high level features in 4 direction and calculate character placement clue
- Filter Gate (FG): Combine the 4 features with the character placement clue



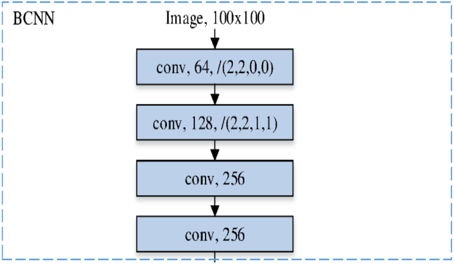
Architecture Overview

- Basal CNN(BCNN): extract low level image features
- Arbitrary(AON): extract high level features in 4 direction and calculate character placement clue
- Filter Gate (FG): Combine the 4 features with the character placement clue
- Attention-based Decoder: predict character sequence from the combined features



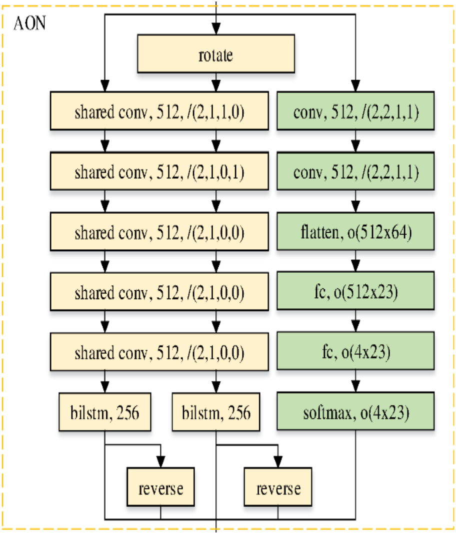
Basal CNN

- Simple stacked CNN
- The output must be square feature maps



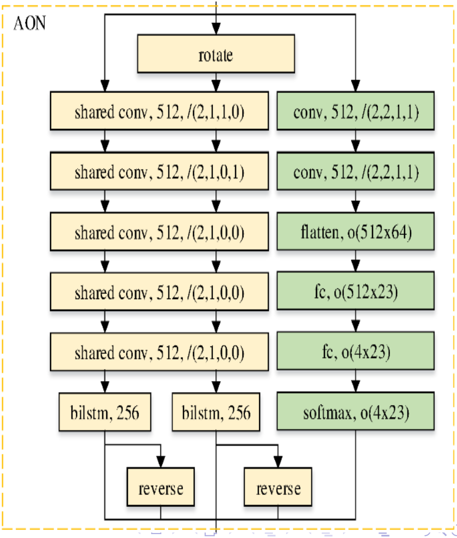
AON I

- Consider "left to right": stacked CNN downsamples the input feature maps from original dimension *HxWxC* to $1 \times L \times D$
- Then feed the feature map to a bidirectional lstm to further encode the feature sequence (keeping the same dimension)
- "right to left" feature is just the reverse of "left to right" feature (which accelerates training convergence)



AON II

- For "Up to down", just rotate the input by 90 degree
- At the end we have 4 $L \times D$ feature maps
- In practice, horizontal and vertical CNN share parameters to avoid unbalanced orientation in the dataset
- The character placement clue network uses CNN-FC to calculate $4 \times L$ weight



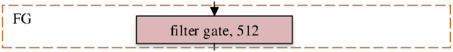
Filter Gate

- Weighted sum of the 4 $L \times D$ features with the $4 \times L$ weights to get a $L \times D$ feature and then activate by tanh function
- For $i = 1, \ldots, L$

$$\widehat{h_i}' = [\overrightarrow{\mathcal{H}_i}, \overleftarrow{\mathcal{H}_i}, \overrightarrow{\mathcal{V}_i}, \overleftarrow{\mathcal{V}_i}] c_i$$

$$\widehat{h_i}' = \tanh(\widehat{h_i}')$$

$$(\overrightarrow{\mathcal{H}_i} : D \times 1, c_i : 1 \times 4)$$



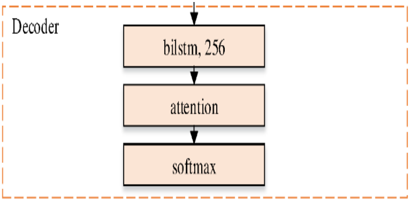
Attention Decoder

- Given previous output y_{t-1}, calculate the decoder input g_t, next state s_t and next output y_t as:

$$g_t = \sum_{j=1}^{L} \alpha_{t,j} \widehat{h}_j$$

$$s_t = RNN(y_{t-1}, g_t, s_{t-1})$$

$$y_t = softmax(W^T s_t)$$



- α_t is a $1 \times L$ vector and can be calculates in different ways. For e.g.

$$\alpha_{t,j} = s_{t-1}^T M h_j$$

Dataset

Name	Size	Irregular	with lexicon
SVT-Perspective	639	yes	50
CUTE80	288	yes	N/A
ICDAR 2015	2,077	yes	N/A
IIIT5K-Words	3,000	no	50,1000
Street View	647	no	50
ICDAR 2003	867	no	50, Full

Trained on 12-million synthetic dataset.

Experiment Result I

Method	SVT-Perspective			CT80	IC15
	50	Full	None	None	None
ABBYY[35]	40.5	26.1	–	–	–
Mishra *et al.*[11]	45.7	24.7	–	–	–
Wang *et al.*[37]	40.2	32.4	–	–	–
Phan *et al.*[28]	75.6	67.0	–	–	–
Shi *et al.*[31]	92.6	72.6	66.8	54.9	–
Shi *et al.*[32]	91.2	77.4	71.8	59.2	–
Yang *et al.*[39]	93.0	80.2	**75.8**	69.3	–
Cheng *et al.*[6]	92.6	81.6	71.5	63.9	66.2
Naive_base	92.4	83.3	70.5	75.4	67.8
STN_base	**94.6**	82.8	68.5	73.7	67.5
Ours	94.0	**83.7**	73.0	**76.8**	**68.2**

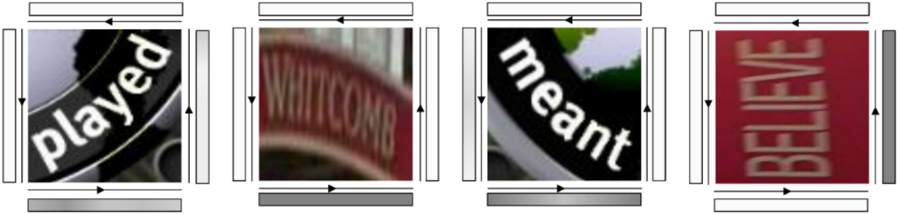
Performance on irregular datasets.

Experiment Result II

Method	IIIT5k			SVT		IC03		
	50	1k	None	50	None	50	Full	None
ABBYY[35]	24.3	–	–	35.0	–	56.0	55.0	–
Wang *et al.* [35]	–	–	–	57.0	–	76.0	62.0	–
Mishra *et al.*[11]	64.1	57.5	–	73.2	–	81.8	67.8	–
Wang *et al.*[37]	–	–	–	70.0	–	90.0	84.0	–
Goel *et al.*[8]	–	–	–	77.3	–	89.7	–	–
Bissacco *et al.*[4]	–	–	–	90.4	78.0	–	–	–
Alsharif [2]	–	–	–	74.3	–	93.1	88.6	–
Almazán *et al.*[1]	91.2	82.1	–	89.2	–	–	–	–
Yao *et al.*[40]	80.2	69.3	–	75.9	–	88.5	80.3	–
Jaderberg *et al.*[16]	–	–	–	86.1	–	96.2	91.5	–
Su and Lu[33]	–	–	–	83.0	–	92.0	82.0	–
Gordo[9]	93.3	86.6	–	91.8	–	–	–	–
Jaderberg *et al.*[17]	97.1	92.7	–	95.4	80.7	98.7	**98.6**	93.1
Jaderberg *et al.*[16]	95.5	89.6	–	93.2	71.7	97.8	97.0	89.6
Shi *et al.*[31]	97.6	94.4	78.2	96.4	80.8	98.7	97.6	89.4
Shi *et al.*[32]	96.2	93.8	81.9	95.5	81.9	98.3	96.2	90.1
Lee *et al.*[22]	96.8	94.4	78.4	96.3	80.7	97.9	97.0	88.7
Yang *et al.*[39]	97.8	96.1	–	95.2	–	–	97.7	–
Cheng's baseline[6]	98.9	96.8	83.7	95.7	82.2	98.5	96.7	91.5
Cheng *et al.*[6]	99.3	97.5	**87.4**	**97.1**	**85.9**	**99.2**	97.3	**94.2**
Naive_base	99.5	**98.1**	86.0	96.9	81.9	98.5	96.5	90.5
STN_base	99.5	97.8	85.9	96.3	80.7	98.5	96.2	89.2
Ours	**99.6**	**98.1**	87.0	96.0	82.8	98.5	97.1	91.5

Performance on regular datasets.

Generated Placement Clues



Text Placement Trends I

- Visualize the model proposed position for each character

Text Placement Trends I

- Visualize the model proposed position for each character
- At each time step we have character placement $\mathcal{C}$ $(4 \times L)$, attention mask α_t $(4 \times L)$

Text Placement Trends I

- Visualize the model proposed position for each character
- At each time step we have character placement $\mathcal{C}$ $(4 \times L)$, attention mask α_t $(4 \times L)$
- Geometrically, the image is divided into $L \times L$ patches, and we try to visualize at each time step, which patch are we looking at

Text Placement Trends I

- Visualize the model proposed position for each character
- At each time step we have character placement $\mathcal{C}$ $(4 \times L)$, attention mask α_t $(4 \times L)$
- Geometrically, the image is divided into $L \times L$ patches, and we try to visualize at each time step, which patch are we looking at
- position distribution

$$dis = (d_1, d_2, d_3, d_4) = \mathcal{C} \odot \alpha_t \in \mathbb{R}^{4 \times L}$$

Text Placement Trends I

- Visualize the model proposed position for each character
- At each time step we have character placement $\mathcal{C}$ ($4 \times L$), attention mask α_t ($4 \times L$)
- Geometrically, the image is divided into $L \times L$ patches, and we try to visualize at each time step, which patch are we looking at
- position distribution

$$dis = (d_1, d_2, d_3, d_4) = \mathcal{C} \odot \alpha_t \in \mathbb{R}^{4 \times L}$$

- d_{1j} measures the importance of the j's column in "left-to-right" feature and d_{2j} measures that of "right-to-left" feature

Text Placement Trends I

- Visualize the model proposed position for each character
- At each time step we have character placement $\mathcal{C}$ $(4 \times L)$, attention mask α_t $(4 \times L)$
- Geometrically, the image is divided into $L \times L$ patches, and we try to visualize at each time step, which patch are we looking at
- position distribution

$$dis = (d_1, d_2, d_3, d_4) = \mathcal{C} \odot \alpha_t \in \mathbb{R}^{4 \times L}$$

- d_{1j} measures the importance of the j's column in "left-to-right" feature and d_{2j} measures that of "right-to-left" feature
- Horizontal position at time step t

$$x = \sum_{i=1}^{2} \sum_{j=1}^{L} j \times norm(d_{ij})$$

Text Placement Trends II